Try to have a properly formatted bibliography!

Summaries:

[1] Xiaoxiao Zhang, Amine Bermak, Farid Boussaid, "Dynamic Voltage and Frequency Scaling for Low-power Multi-precision Reconfigurable Multiplier", in Proc. of 2010 IEEE International Symposium on Circuits and Systems, pp. 45-48, Paris, 2010

In this paper, the author introduced the same idea in reconfigurable multiplier design. The proposed multi-precision multiplier enables voltage and frequency scaling for low power operation, while still maintaining full throughput. They applied several strategies in low power design,

1. Dynamic voltage scaling (DVS) saves energy by scaling down the voltage supply when the processor is not fully loaded.
2. Frequency scaling via VCO fed by lower voltage.
3. Choice of Partial Products Compression Topology
4. Various algorithms and topologies are explored to obtain high performance while no silicon area or power overheads.

The idea of DVS and Frequency Scaling can be used in our project. In this paper, voltage dithering is utilized to provide near-optimum dynamic voltage scaling with much less overhead. This can be also applied as our dynamic power management.

[2] Z. Lu , Y. Zhang , M. Stan , J. Lach and K. Skadron "Procrastinating voltage scheduling with discrete frequency sets", *Proc. Des. Autom. Test Eur.*, pp.456 2006

This paper introduced an optimal procrastinating DVS for systems with discrete frequency sets. By analyzing the system and solving the optimal problems considering the overheads, they demonstrate that the algorithm is efficient and can be applied in various practical systems.

We can try to utilize the idea of procrastinating voltage scheduling in our project, since we will have tree adders and stage-by-stage circuitry. The conclusion for a given deadline, a small number of carefully selected frequencies may be sufficient to form a good schedule achieving energy savings comparable to that using all efficient operating points provides us a guideline in dynamic voltage scaling and frequency scaling.

[3] Xiaoxiao Zhang, Amine Bermak, Farid Boussaid, “Power Optimization in Multipliers Using Multi-Precision Combined with Voltage Scaling Techniques”, in 1st Int'l Symposium on Quality Electronic Design-Asia. 2009

This paper presented a low-power high-performance multi-precision multiplier. Dynamic input range detection was employed to examine the multiplier’s operands. Voltage scaling and algorithm optimization were also employed. Eventually, results showed that the multiplier achieved up to 75% power reduction at a cost of 10% silicon area overhead.

It employed a similar idea to ours, which is to detect the input range and control the bit length. While in our design, we will divide the multiplicand and multiplier into k groups, respectively. Then all the following operations will focus on the small blocks. Maybe in this way we can achieve lower power and higher performance. Another difference might be that we will develop procrastinating voltage scheduling in our design, resulting in higher energy efficiency.

[4] Jiun-Ping Wang, Shiann-Rong Kuang, Yuan-Chih Chuang, ”Design of Reconfigurable Low-Power Pipelined Array Multiplier” Communications, Circuits and Systems Proceedings,Volume 4, Page(s):2277 - 2281, 25-28. June 2006

This paper presented a low-power reconfigurable signed pipeline array multiplier. Partially guarded computation and truncated multiplication techniques are employed in their design to reduce power consumption. Control signal (CS) is used to control the operation of MSP based on the width of sign extension bits. We can probably utilize this concept in our design. Not only do we control the LSB part, but we can also control the MSB part to reduce the power further.

Project Progress:

Idea Formation

The idea of implementing an imprecise low power block array multiplier has been solidified in our group. And we are continuously thinking about new ideas to improve our design. The idea states that we can first divide an n*n bits multiplier into k*k groups of multipliers. Each group of multiplier should have n/k bits. Now since we can have finer grain control over each much smaller bits multiplier, we can easily turn off some of the lower significant multipliers to speed up the process and reduce the power. At current stage, the preliminary design idea has been verified to be suitable for the project and the implementation. However, we still need to find some more ideas on how to further improving and optimizing our design. And there are multiple ways to do this including reading more literatures and talking to different people including professors.

Literature Research

At this moment, our group has explored significant amount research publications in power aware and reconfigurable multipliers. We found out that several papers that were published in the last two years have very similar ideas to ours. However, there are still a lot of differences and we believe that by further implementing and optimizing our design, we will have a much faster and power efficient design then current implementations. In order to get more ideas on how to implement certain features such as voltage and frequency scaling, we still need to read significant more publications.

Algorithm Validation and Simulation

At present stage, we have written the algorithm for our low power multiplier in Python language. Through large amount of random data simulation, the algorithm design has been successfully verified. The algorithm implemented in python is exactly the same as the design that will be implemented in VHDL and RTL level since it directly operates on bits. During the simulation, multiple parameter set has been chosen for the simulation based on the likelihood of that parameter set could be potentially implemented in the real design. Each simulation has 10k random inputs for various lengths of bits at different settings of precision. And the output of these simulations is the error magnitude and the discrepancy percentage in magnitude. The purpose of these simulations is to further verify the design from the algorithm perspective and give us an idea of what parameter settings might give the best precision result. Therefore, these simulations are very necessary and can be really helpful for the lower level design. We will continue developing the VHDL and lower level circuits of our design, and conduct large amount of random data simulations. Eventually, we would like to have the entire multiplier circuits built in transistor level, and have the power and delay simulated.

Remaining Tasks:

Tasks Before Proposal:

There is one week between the date of design review and the proposal due date. In this one week , we plan to do the following:

a. Implement a VHDL circuit of 8*8 bits lower power block array multiplier
b. Simulate the design in terms of logical function and timing delay
c. Compare preliminary results to the state of art
d. Conduct power and timing simulation if possible in the gate level
e. Optimize the combinational delay by introducing a control block that could turn off each individual multiplier blocks.
f. Create a detailed and well-structured design proposal
g. Continue exploring the literature

Tasks After Proposal:

After the proposal is submitted, we should have a very simple and 8 by 8 bit low power block array multiplier available. And more in depth research on the project should be conducted. There will be a lot of tasks following the proposal as some of them are listed as following:

a. Extend our multiplier to higher number of bits including 16*16, 24*24, and 32*32. Eventually we should have all this multipliers implemented in the transistor level.
b. Compare the design to state of art
c. Introduce dynamic control logic that will automatically turn off some the multipliers according to the input values. For example, if one operand of input to one small block is all '0', then either that entire row or column mulitipliers will all be zeros, therefore, we can choose to turn off them automatically to save power.
d. Apply the idea of voltage procrastination
e. Implement these multipliers in transistor level. And optimize the performance using various VLSI techniques.
f. Conduct significant amount of power, speed and accuracy simulations. Compare the results to the state of art design. If not better than state of art, then we need to continue improve our design.
g. Synthesize our design of the multiplier and make it ready for manufacturing.

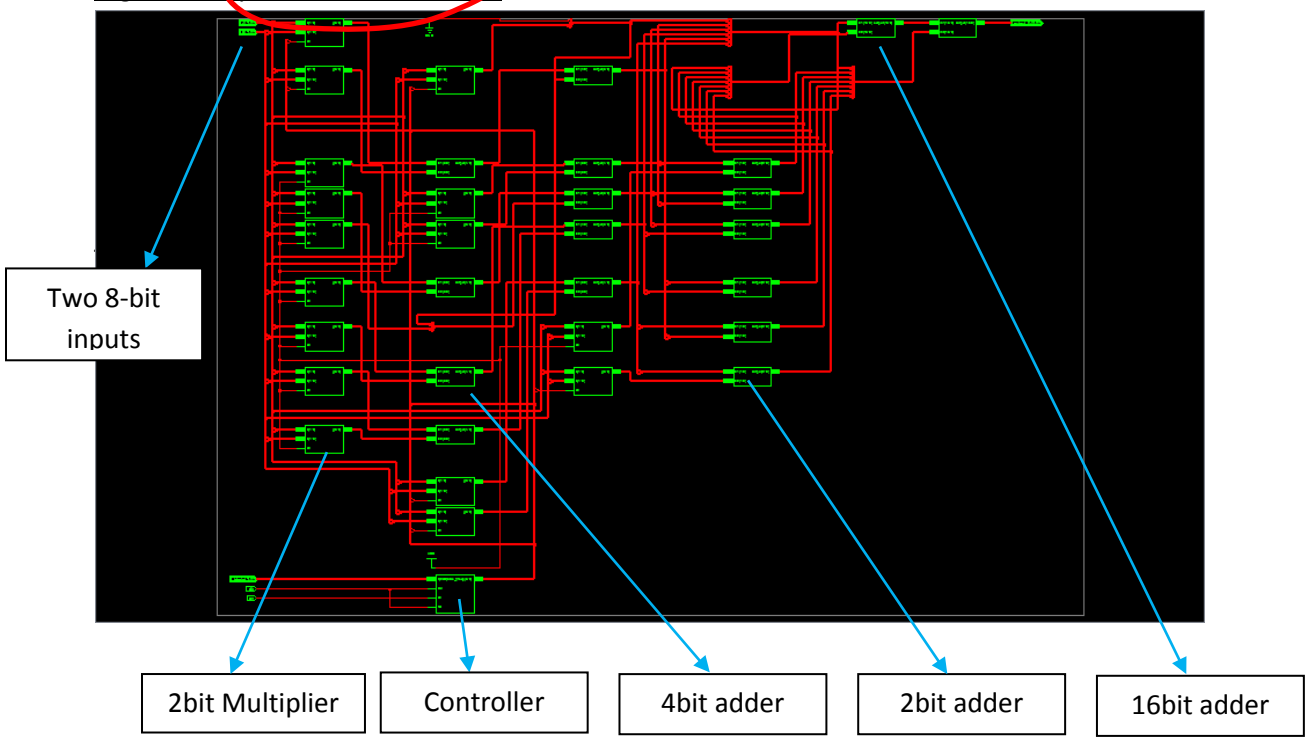
Simulations:

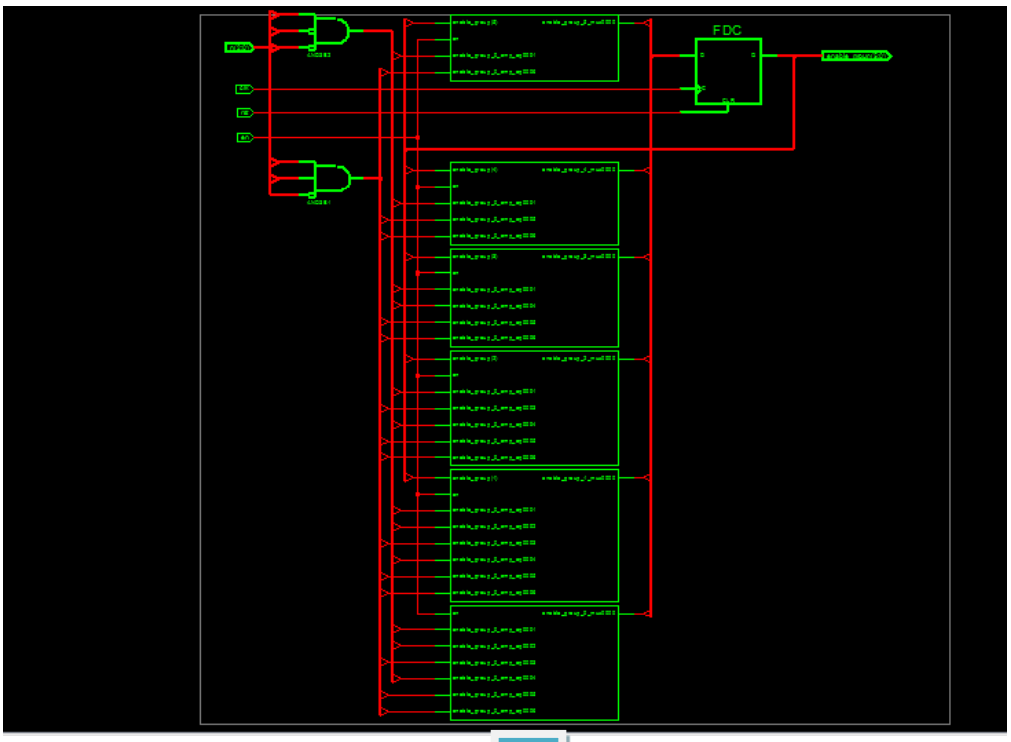
Truncation Error Simulation Result:

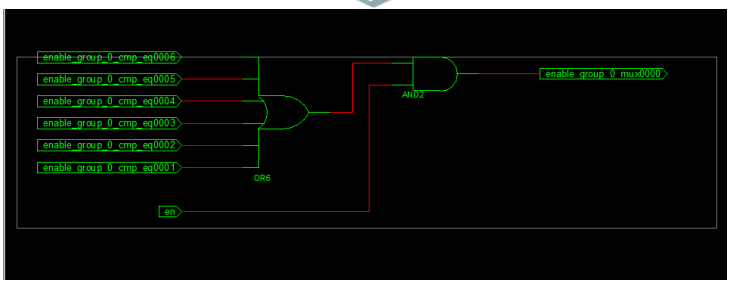
The table below gives the simulation result on the truncation error. The data is collected by running 10k random input simulation on each combination of N,K,and Xb. The mean absolute error and root mean square error are given. N represents the total number of bits, K is the number of blocks for each multiplicand (K*K blocks in total) and Xb are the number of truncated blocks. As we can see from the result, certain parameters settings give better accuracy than others for a fixed number of bits. When it is an 8 by 8 bit multiplier, we can see when K = 4 and xb = 1 the mean absolute error is very small compared to when K = 2. This enables us to explore the best parameter set for the multiplier design once we also have the data on power consumption and delay. After this design review we plan to give a more comprehensive simulation and come up with better estimation logics instead of truncating blocks directly to reduce the error magnitude as much as possible while keep the power consumption low.

N	K	Xb	Mean Absolute Error(MAE)	Root Mean Square Error(RMS)
8	2	1	76	104.9599071074
8	4	1	3	4.9470496258
8	4	3	21	32.0013952821
16	4	1	77	105.4394466033
16	4	3	1870	334.5141218245
24	8	1	17	24.2533750229
24	8	3	214	301.0916654443
24	8	6	2896	440.8787722266
32	8	1	77	105.8042877203
32	8	3	1891	445.4737344670

High Level Synthesized Schematic:




FDC


enable_group_0_cmp_eq0006
enable_group_0_cmp_eq0005
enable_group_0_cmp_eq0004
enable_group_0_cmp_eq0003
enable_group_0_cmp_eq0002
enable_group_0_cmp_eq0001
en
OR6
AND2
enable_group_0_mux0000

Synthesis Report

```
#      LUT2                : 39
#      LUT3                : 38
#      LUT4                : 72
#      MULT_AND            : 16
#      MUXCY               : 54
#      MUXF5               : 1
#      XORCY               : 37
# FlipFlops/Latches        : 6
#      FDC                 : 6
# Clock Buffers            : 1
#      BUFG                : 1
# IO Buffers               : 37
#      IBUF                : 21
#      OBUF                : 16
=========================================================================
```

Device utilization summary:

```
---------------------------
Selected Device : 4vlx100ff1148-10
 Number of Slices:                 85  out of  49152     0%
 Number of Slice Flip Flops:        6  out of  98304     0%
 Number of 4 input LUTs:          151  out of  98304     0%
 Number of IOs:                    38
 Number of bonded IOBs:            37  out of    768     4%
 Number of GCLKs:                   1  out of     32     3%
=========================================================================
```

VHDL simulation:

The VHDL simulation shows that when ctrl signal changes the result reflects the truncated results. When ctrl goes smaller, the error is larger. However, there is a slight error in the simulation where at the beginning when ctrl = '111', it should give the correct result. And there seems to have a delay. But, after that, the simulation shows as expected.